Project document - FoundrMatch

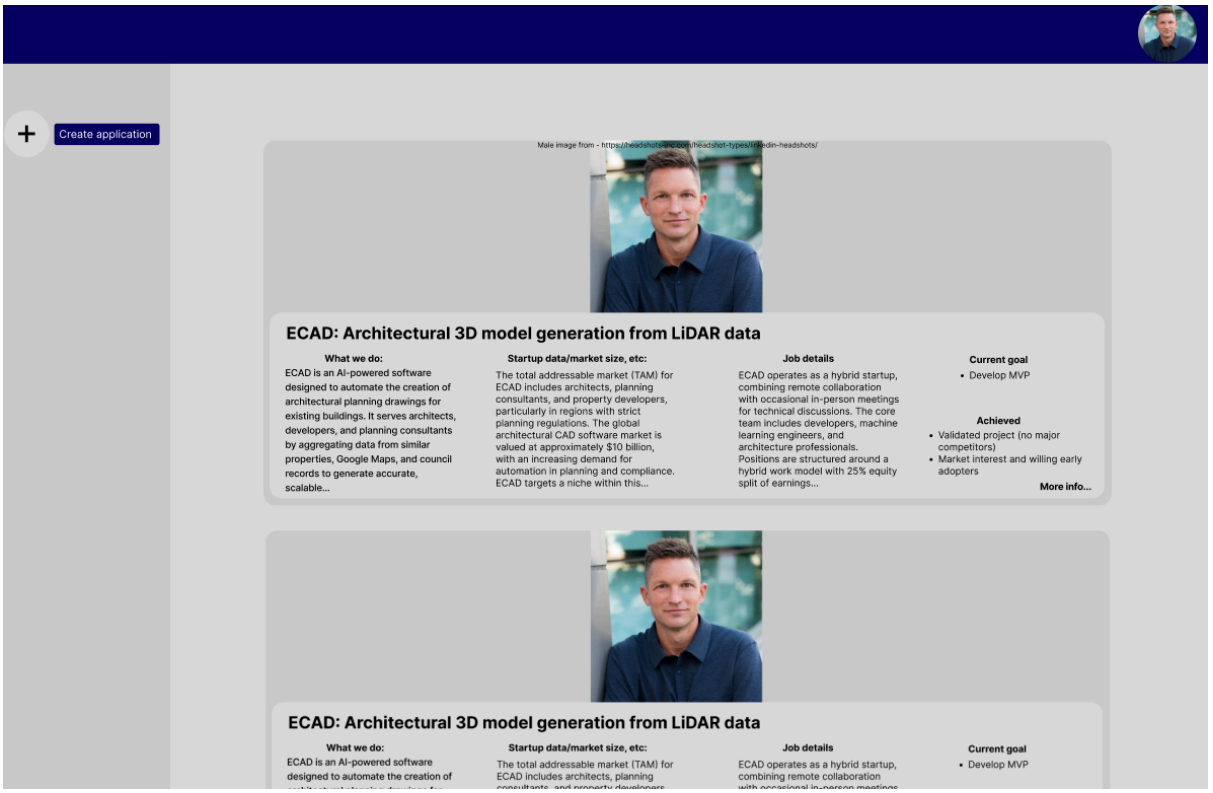
Problem:

Early stage startups - Hard to hire on equity, companies that are ‘bootstrapped’ and have no funding from outside investors struggle to offer salaries, therefore hire on equity and struggle to find applicants interested in joining. These founders usually have to reach out to programmers directly through LinkedIn.

Programmers/Computer Scientists - Unemployed, worm eating and homeless, I think that there still are jobs in the market that yield returns, but don’t have much exposure as typical salary based job offers.

Solution:

Software to connect early-stage startups (new companies) that are hiring on equity based roles (shares of their company, and percentage of earnings) to programmers and other CompSci people that are looking for job opportunities.



Features:

- **Sign up/create account (top right of image button)**

The site is ‘online’, you don’t have to create an account just to see job applications. Although in order for a founder to create a job application for their startup, they should sign up with their email.

Use **Flask** (Python web framework) to handle user authentication.

- Store user credentials securely using **Flask-Login** and **Werkzeug** for password hashing.

- Use **SQLite** or **PostgreSQL** for a simple user database.
- Implement an API route (`/register` and `/login`) to handle signups and logins.
- Use **WTForms** for form validation (checking if fields are filled correctly).
- If using **Flask**, store sessions using Flask's built-in session management or **JWT tokens** for authentication.
- Frontend: Use **HTML/CSS/JS** for the signup form, **Bootstrap** for styling.

- **Algorithm to determine visibility of job applications**

I do not want to lower quality/less mature startups to have as much visibility relative to a startup that is more mature and can provide better conditions and a more professional approach when working with. For example, an 18 year old with a 'big idea' that has had no interest from a market and thinks he is going to be Elon Musk in the next couple of years should not have a job application that is visible, compared to a startup from someone with an education, has market interest and even an already developed MVP with adoption.

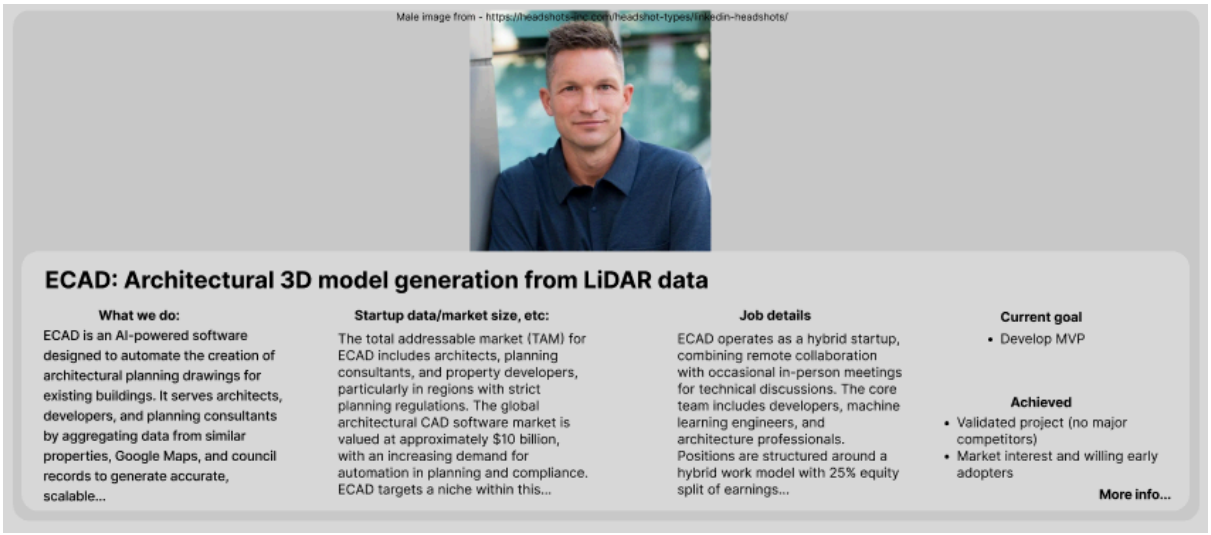
On the page for 'sign up'/'create account', users will be required to input their startup details if they want to create a job application.
Their startup details, collected from the questions below, should be able to determine the 'maturity level' of the startup.

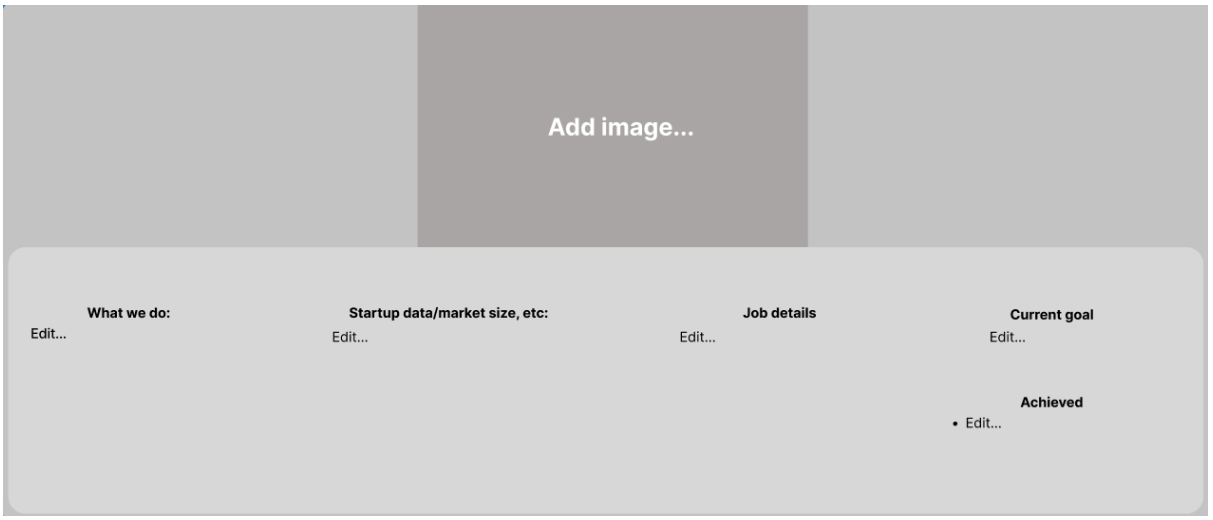
- Age
- Level of education preferably with some verification
- Already achieved in startup
- Previous experience in professional projects
- Years of experience in previous projects

These weightings assigned to different levels of responses should determine how visible a job application is to programmers. And when I say 'visible' I am speaking about their ranking/position from top of list of job applications on the home page (first image).
Specific weightings for determining job visibility is not known yet (yet to think about it) (I can do it if you do not want to)

- Assign a **visibility score** to each job based on weighted criteria (startup maturity, founder's background).
- Define weightings for:Founder's **education level** (higher weight for degrees, professional experience).
    - **Startup stage** (MVP launched > market interest > idea stage).
    - **Founder's past experience** in startups/projects.
- Use **SQL queries** to store visibility scores in the database (`visibility_score` column).
  Sort jobs by visibility score before displaying them (`ORDER BY visibility_score DESC`).

- **Create job application page**





On the home page, button on the left of the home page ‘Create application’, users should go to a different page → They will input details on a similar template to above, directly editing what their job application will look like. Realise that the template is the same as the actual job application.
THESE ARE NOT THE ONLY QUESTIONS, the box should expand into more questions, which are required to be input:

- Name
- Personal details, story and goals
- What their startup does
- Startup data, market sizes, TAM, etc
- Their current position in their startup (idea/concept, market interest,major market interest, MVP, initial adoption, etc)
- Working details (remote, hybrid, with required specific equity split/running salary)

These questions should be on the job application directly, although these questions do not

play a role in determining the visibility of the job application/no effect on the algorithm. These are just to directly explain everything to the programmer/job applicant.

- Form using **Flask-WTForms** to collect job details.
- Store job applications in a **PostgreSQL/SQLite** database.
- Fields stored:
    - Name, personal details, startup info, market size, current position, work details.
- Use Flask's `request.form.get()` method to collect user input and insert into the database.
- Ensure form validation (e.g., required fields, correct data types).
- Display submitted jobs on `/jobs` page dynamically.

Applying to jobs (as a programmer)

If programmer sees a job offer at a startup that they would be interested in working in. Press 'more details…' on job application (button shown in first image) >> they will see a button to apply >> they can retrieve contact details of founder/startup/a phone number, email, etc, and contact them directly about the job position.

(I prefer this approach of the job applicant contacting the startup, opposed to startups/companies contacting the job applicants as it acts as a barrier to entry to job application, filtering out candidates that are not committed/serious about the job position, ensuring companies have higher quality candidates)

- Display "More details" button using **HTML + Flask templates**.
- Clicking it fetches full job details from the database (`/job/<job_id>` route).
- Store startup contact details in the database and display them on job listings.
- Add a "Copy Contact Info" button (basic **JavaScript event listener** to copy email/phone).
- No application form—users contact the startup manually via displayed details.

Viewing Job Applications (Without Account)

- Store job applications in a **PostgreSQL** or **SQLite** database.
- Create a `/jobs` API route in Flask that queries and returns jobs from the database as JSON.
- Display job postings using **Jinja templates** in Flask or fetch via **AJAX** for a dynamic experience.
- Jobs should be sorted by a visibility score (see the algorithm section).