I & E

Project Proposal:

HOMIES

Team 4: Abeera, Chloe, Lamia, Qaisar

Homies

Harmony in Every Home

We solve the problem of confusion and conflict over household chores in shared homes for roommates, couples, and families by providing an app that allows users to assign, track, and manage tasks transparently, ensuring fairness and accountability.

Overview

1. Situation & Problem
2. BMC: Revenue Streams, Prices
3. Solution & Value Proposition
4. Functionalities & Screens
5. Code & Demo
6. Manual Prototype
7. Application: Technology & Tests
8. Customers: Acquisition & Retention
9. Users List
10. Market Study (Users & Customers)
11. Interviews
12. Questionnaire
13. Market Study (Users & Customers): Sources
14. Market Study (Competitors)
15. Market Study (Competitors): Sources
16. Detailed Analysis of User Behaviour
17. User Feedback & Analysis: Manual Prototype
18. User Feedback & Analysis: Application
19. Financial Plan
20. Conclusion
21. Presentation Responsibility

Situation & Problem

Situation

– **Qaisar**: university student living with 3 roommates, struggles to coordinate household chores in a shared apartment.

– Each roommate has a different schedule, so tasks like cleaning, taking out the trash, or washing dishes often get delayed.

– Occurrence: **Daily / Weekly chores**, recurring issues multiple times per week.

Problem

– **Confusion** over who is responsible for what task.

– Tasks are forgotten or **unevenly distributed**, leading to **frustration** and **tension** among housemates.

– Lack of transparency causes **repeated arguments** and **decreased motivation** to do chores.

BMC: Revenue Streams & Prices

Solution & Value Proposition

Solution

Homie, an application that allows users to:

• Create a shared household space and add members

• Assign tasks with deadlines

• Track completed tasks and contributions

• Send reminders and notifications

• Reassign or request help for chores

Value Proposition

Time saved: Quick assignment and tracking reduce back-and-forth communication (~10 min/week saved per person).

Mind peace: Transparent tracking reduces arguments and frustration.

Fairness: Everyone’s contributions are visible, promoting accountability.

Consistency: Recurring reminders ensure chores are done on time, keeping the home organized.

Functionalities

Must-Have

- **Create a household space**
- **Add members or join household via link or QR code**
- **Create a task**
- **Assign task to a household member**
- **Add a deadline (specific date)**
- **Mark task as done**
- **Reassign task / request help**
- **Activity log** (list showing who completed which tasks)
- **Calendar views:** daily / weekly / monthly
- **User account creation + login**
- **Email notifications** for reminders

Nice-to-Have

- **Point or reward system**
- **Task categories** (cleaning, cooking, shopping…)
- **Color-code for each member**
- **Add a time for a task**
- **Mark urgent vs non-urgent tasks**
- **Delay/postpone a non-urgent task**
- **Comment section under each task**

Screens

- **Dashboard page** → summary of today’s tasks
- **Task creation form** → name, deadline, assignee
- **Calendar page** → day/week/month view
- **Household page** → members and invitation link
- **Task details page** → comments, urgency, mark as done
- **Activity log page** → list of completed tasks
- **Profile / settings page** → notifications, household options

Code & Demo

Implementation Overview

- The prototype interfaces and screens were built using **TypeScript** with a modular structure.
- Each screen is organized into reusable components for easy iteration and future scalability.
- The code reflects the app's core logic at a conceptual level-handling navigation, user input, state updates, and simulated data retrieval.

Code Example (Simplified)

```ts
// Basic types
type Task = {
  id: string;
  title: string;
  completed: boolean;
  deadline: Date;
  assignedTo?: string;
};

type Member = {
  id: string;
  name: string;
};

// Simple task list component
function TasksList(props: {
  tasks: Task[];
  members: Member[];
  onToggle: (id: string) => void;
  onDelete: (id: string) => void;
}) {
  const getMemberName = (id?: string) =>
    props.members.find((m) => m.id === id)?.name ?? "Unassigned";

  return (
    <div>
      {props.tasks.map((task) => (
        <div key={task.id}>
          <input
            type="checkbox"
            checked={task.completed}
            onChange={() => props.onToggle(task.id)}
          />

          <strong>{task.title}</strong>

          <span> — {getMemberName(task.assignedTo)}</span>

          <span> — {task.deadline.toDateString()}</span>

          <button onClick={() => props.onDelete(task.id)}>Delete</button>
        </div>
      ))}
    </div>
  );
}
```

Code & Demo

Key Features Demonstrated

- **User Registration Flow**
  Simple form validation, error handling, and input state management.

- **Appointment Scheduling Flow**
  Allows the user to choose a day, pick a time slot, and confirm an appointment.
  All implemented with a clean TypeScript component architecture.

- **Dashboard Mock Logic**
  Components simulate:
  - Upcoming appointment display
  - Past appointment list
  - Quick actions for booking or canceling

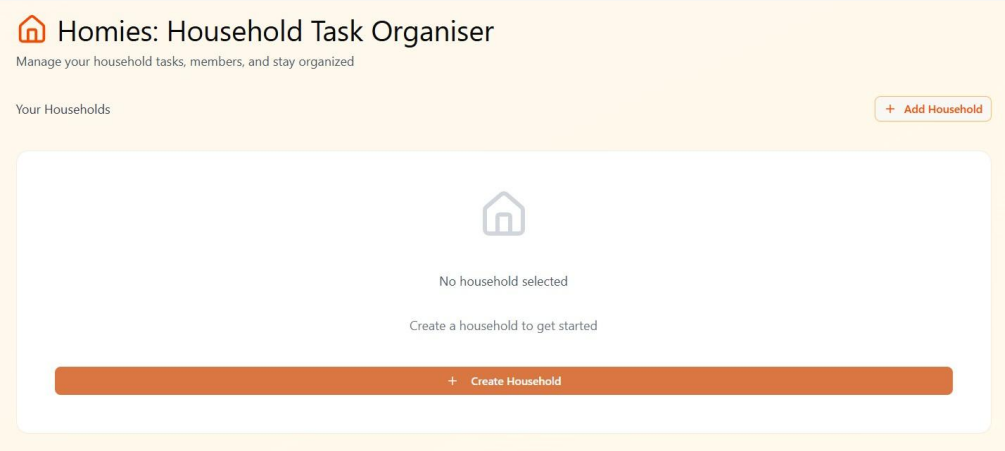
Demonstration Summary

- The prototype runs through the full booking cycle:
  - **Home → Choose Service → Pick Date/Time → Confirm**
  - **View appointments**
  - **(Simulated) modify or cancel**
- The aim is to illustrate core app logic before implementing the backend and final design.

Manual Prototype

Why a Manual Prototype?

- To validate the concept early.
- To test user flow and usability with minimal development investment.
- To communicate the idea clearly during development planning.



Manual Prototype

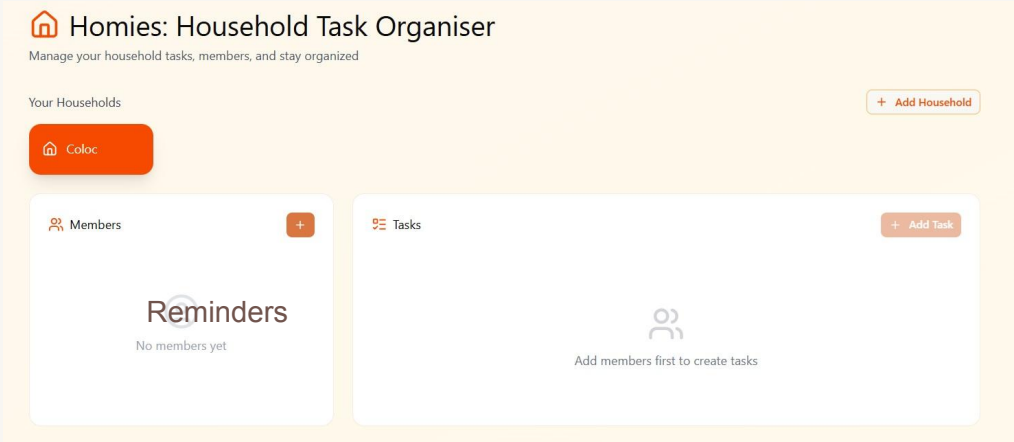
Prototype Structure

The manual prototype contains all main screens of the planned application,

representing:

- Onboarding
- Home/dashboard
- Task Assignment workflow
- Tasks details
- Completion

and

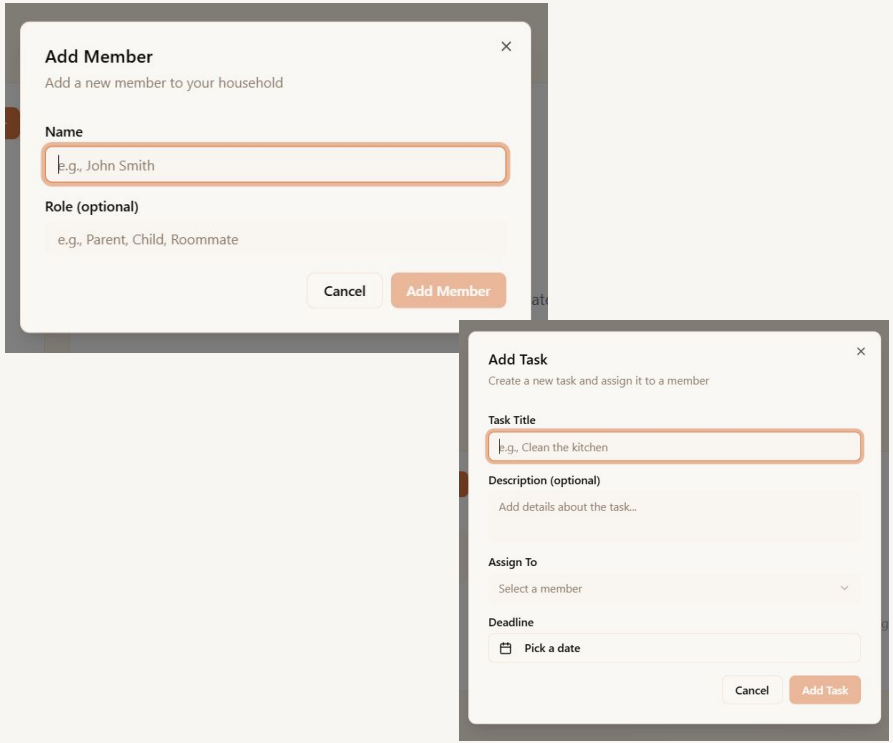


These layouts show exactly how a user will move from one step to the next.

Manual Prototype

How It Works (Concept Flow)

- A household is created.
- Members are added to the household.
- Member Adds/confirms a task.
- Backend generates:
  - optional calendar event data
- User receives reminders at scheduled times.
- If user changes or cancels the appointment:
  - notifications are updated or deleted accordingly



Application – Technology Chosen

- **Web application** accessible via browser on desktop or mobile
- To developed using **modern web technologies** (TypeScript + Next.js)
- Focus on **fast, responsive, and collaborative experience**
- Designed to support **real-time updates**,modifications, and user friendly dashboards

Why a Web App?

Accessible Anywhere: Works on desktops, laptops, tablets, and mobile browsers hence no installation needed.

Cross-Platform: All household members can access the app regardless of device type.

Easy Updates: New features or bug fixes are instantly available to all users.

Centralized Data: All tasks, logs, and notifications are synced in real time for everyone.

Scalable & Future-Ready: Can later add features without major changes.

Customer Acquisition

Acquisition:

- Target **shared households and student dorms**
- Promote via **social media, university campaigns, referrals**
- **Simple onboarding** for multiple members

Customer Retention

Retention:

- **Transparent logs & dashboards** fairness & accountability
- **Notifications & reminders** for task completion
- **Gamification:** points, badges
- **Personalization:** color coded members, task categories, urgency labels

Customer Acquisition & Retention

CRM VIEW:

- **New Users:** Help them get started quickly → onboarding support
- **Active Users:** Keep them motivated → rewards & engagement incentives
- **Passive Users:** Encourage participation → gentle reminders & tips
- **Dormant Users:** Bring them back → re-engagement notifications
- **Household Admins:** Manage the household → assign tasks & encourage members

Customer Acquisition & Retention

Customer Acquisition

- **Shareable link / QR code to join a household**
- **Social media presence**
- **Onboarding tutorial**
- **Referral system** (optional)

Retention Strategy

- **Reminder notifications through email**
- **Email reminders for overdue tasks**
- **Comments under tasks**
- **Weekly summary delivered by email** (optional)
- **Household stats**

Accounting & Analysis

- **Monthly Active Users**
- **Monthly Active Households**
- **Task completion rate**
- **New accounts created per week**
- **Retention after 1 week, 4 weeks, 12 weeks**
- **Churn rate** (how many users stop using the website)
- **Bug reports & feedback form responses**

Users List

Market Study (Users & Customers)

Interviews

Questionnaire

Market Study (Users & Customers): Sources

Market Study (Competitors)

Market Study (Competitors): Sources

Detailed Analysis of User Behaviour

User Feedback & Analysis

Manual Prototype

User Feedback & Analysis

Application

Financial Plan

Conclusion

Presentation Responsibilities