

**Dr. A.P.J Abdul Kalam Technical University**  
**Lucknow, Uttar Pradesh**

# **Manobal**

**( A Personality Development App )**

**Mentor Name : Prof. Hemant Bhardwaj**

**Course : Bachelor of Technology in Computer Science Engineering**

**Authors : Khushi Chaudhary (2102310100055), Nitish Kumar (2102310100070),  
Prabhat Chaudhary(2102310100073), Prince Kumar(2102310100075)**

# Abstract

*Manobal* is a comprehensive mobile application designed to facilitate personality development by helping individuals overcome challenges such as low confidence, poor communication skills, and stage fear. Rooted in the **OCEAN** model of personality traits, the app adopts a structured, user-centric approach to foster holistic personal growth. It provides personalized development plans, interactive learning modules, and real-time progress tracking tools to support users in their journey toward self-improvement.

The platform begins with an initial assessment, where users answer a series of questions to evaluate their current personality traits, strengths, and areas for improvement. Based on these insights, the app curates customized development pathways to enhance confidence, communication abilities, and social adaptability.

Key features of *Manobal* include:

1. **Interactive Learning Modules** – Exercises focused on public speaking, interpersonal skills, and effective communication.
2. **Gamified Learning** – Daily challenges, quizzes, and reward-based engagement to encourage consistent participation.
3. **Progress Tracking** – Visual dashboards for monitoring personal achievements and long-term growth.
4. **Community Engagement** – A collaborative space where users can share experiences, participate in group activities, and practice acquired skills in a supportive environment.

Designed with an intuitive and accessible interface, *Manobal* ensures ease of use for individuals from diverse backgrounds. By integrating psychological insights with technology, the app aims to create a safe, judgment-free space for self-improvement, empowering users to develop essential life skills with confidence.

---

# Introduction

Personality Development Plays a vital role in shaping an individual's confidence, communication skills, and overall social adaptability. Many individuals face challenges such as low self-esteem, stage fear, and ineffective communication, which hinder both personal and professional growth. Overcoming these obstacles requires a structured and personalized approach that fosters continuous self-improvement.

This research introduces *Manobal*, a mobile application designed to help individuals enhance their personality by addressing key challenges through **interactive learning modules, structured development plans, and progress tracking tools**. The application is based on the **OCEAN model of personality traits**, which categorizes personality into five dimensions—Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism. By leveraging this psychological framework, *Manobal* provides users with an initial self-assessment to identify strengths and areas for improvement, enabling a tailored learning experience.

The app Incorporates **engaging activities, gamified challenges, and a supportive community environment** to ensure a motivating and effective learning process. Users participate in **daily tasks, self-improvement exercises, and interactive sessions** to gradually build confidence, improve communication skills, and overcome personal limitations. Additionally, a **progress tracking feature** helps users monitor their growth over time, reinforcing long-term behavioural improvements.

The objective of this study is to evaluate the impact of structured personality development programs and examine how mobile-based solutions can effectively support individuals in enhancing their self-confidence and interpersonal skills. By providing a **safe, judgment-free space**, *Manobal* aims to empower users to achieve holistic personal growth.

---

# Methods

## **Design :**

The study employed a **between-subjects design**, where participants were assigned personalized learning modules based on their initial assessment.

The independent variable was the **structured personality development program**, divided into three levels:

1. **Basic Level** –Personality Assessment, Authentication.
2. **Intermediate Level** – Progress Tracking, Task Generation .
3. **Advanced Level** – Real-world application, including group discussions and feedback sessions, With Community Support.

The dependent variable was **personal growth and skill improvement**, measured through:

- Self-reported confidence levels before and after using *Manobal*.
- Engagement rates with different modules.
- Task performance, including communication exercises and public speaking tasks.

Additional variables, such as **community engagement, and task completion rates**, were also recorded.

## **Development :**

Participants interacted with the *Manobal* mobile application, which provided:

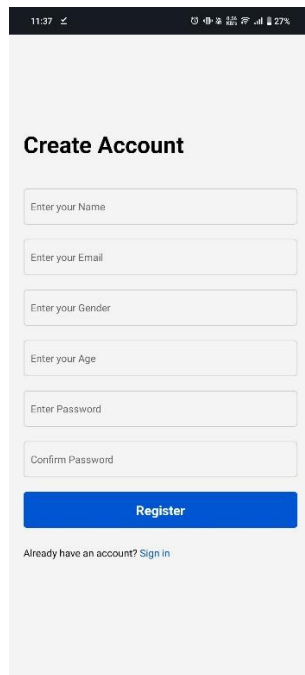
- **Landing Page** of our App to onboard users.



- **Authentication** of our app is based on JWT ( JSON WEB TOKEN ) which is controlled by backend.

➤ Algorithm to Authenticate Users ( JavaScript Middleware ) :

```
const authMiddleware = (req, res, next) => {
  const token = req.header('manobal'); // Bearer <token>
  if (!token) {
    return res.status(401).json({ message: 'No token provided, access denied' });
  }
  try {
    const decoded = jwt.verify(token, process.env.JWT_SECRET); // Replace with your secret key
    req.user = decoded; // Attach user info to the request object
    next(); // Pass control to the next middleware or route handler
  } catch (err) {
    return res.status(401).json({ message: 'Invalid token' });
  }
};
```



**Create Account**

Enter your Name

Enter your Email

Enter your Gender

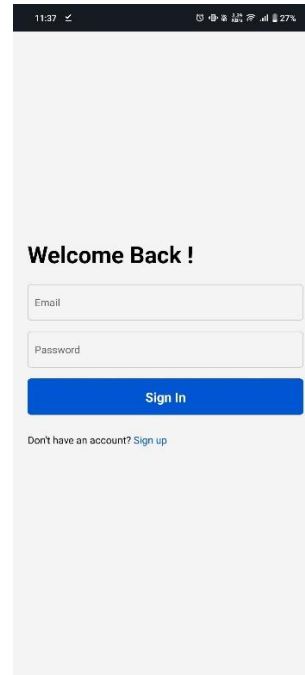
Enter your Age

Enter Password

Confirm Password

**Register**

Already have an account? [Sign in](#)



**Welcome Back !**

Email

Password

**Sign In**

Don't have an account? [Sign up](#)

- **Personalized Development Plans** based on the OCEAN personality model.

**OCEAN Model** : OCEAN is a personality model representing Openness, Conscientiousness, Extraversion, Agreeableness, and Neuroticism, defining human traits and behavioural tendencies scientifically.

1. **Openness** – Creativity, curiosity, and willingness to explore new experiences, ideas, and perspectives, fostering adaptability and intellectual growth.
2. **Conscientiousness** – Organization, responsibility, and self-discipline, ensuring goal-directed behaviour, reliability, and efficiency in personal and professional tasks.
3. **Extraversion** – Sociability, enthusiasm, and assertiveness, influencing energy levels, interpersonal interactions, and preference for engaging with social environments.
4. **Agreeableness** – Compassion, cooperation, and trustworthiness, shaping positive relationships, empathy, and the ability to work harmoniously with others.
5. **Neuroticism** – Emotional stability versus stress sensitivity, determining resilience, mood fluctuations, and susceptibility to anxiety, depression, or emotional distress.

➤ Algorithm to analyse the personality of a user ( JavaScript ) :

```
const assessment = async (req, res) => {
  try {
    const { answers } = req.body;

    if (!answers || !Array.isArray(answers)) {
      return res.status(400).send({
        success: false,
        message: 'Invalid input: answers must be an array.',
      });
    }

    const traits = {
      Openness: [],
      Conscientiousness: [],
      Extraversion: [],
      Agreeableness: [],
      Neuroticism: [],
    };
  }
}
```

```

for (const answer of answers) {
  const { id, _id, answer: score } = answer;

  if (score < 1 || score > 5) {
    return res.status(400).send({
      success: false,
      message: `Invalid score for question ID ${id}: must be between 1 and 5.`,
    });
  }

  const ques = await questionModel.findOne({ id });

  const trait = ques.trait;

  if (trait) {
    traits[trait].push(score);
  }
}

const result = {};
for (const trait in traits) {
  const scores = traits[trait];

  let sum = 0;

  for (const score of scores) {
    sum += score;
  }

  const average = sum / scores.length;
  result[trait] = average.toFixed(2);
}

const userEmail = req.user.email;
const scoreExist = await scoreModel.findOne({ userEmail });
if (scoreExist) {
  await scoreModel.deleteOne({ userEmail });
}
const userScore = new scoreModel({ userEmail, scores: result });
await userScore.save();

```

```

await userModel.updateOne({ email: userEmail }, { isAssesmentDone: true });

return res.status(200).send({
  success: true,
  message: 'Personality assessment completed successfully!',
  data: result,
});
} catch (error) {
return res.status(500).send({
  success: false,
  message: error.message,
});
}
};

```

11:37 2G 4G 5G 6G 7G 8G 9G 10G 11G 12G 13G 14G 15G 16G 17G 18G 19G 20G 21G 22G 23G 24G 25G 26G 27G 28G 29G 30G 31G 32G 33G 34G 35G 36G 37G 38G 39G 40G 41G 42G 43G 44G 45G 46G 47G 48G 49G 50G 51G 52G 53G 54G 55G 56G 57G 58G 59G 60G 61G 62G 63G 64G 65G 66G 67G 68G 69G 70G 71G 72G 73G 74G 75G 76G 77G 78G 79G 80G 81G 82G 83G 84G 85G 86G 87G 88G 89G 90G 91G 92G 93G 94G 95G 96G 97G 98G 99G 100G

Hi, **Tester4**

Complete Your Assessment

Do you often volunteer or offer help to others without being asked?

1

2

3

4

5

< >

Submit Assessment

- **Progress Tracking** through a visual dashboard for individual Traits and Overall

➤ Algorithm for the Progress Tracking ( JavaScript ):

```

const score = async (req, res) => {
  try {
    const userEmail = req.user.email;
    const userScore = await scoreModel.findOne({ userEmail });
    if (!userScore) {

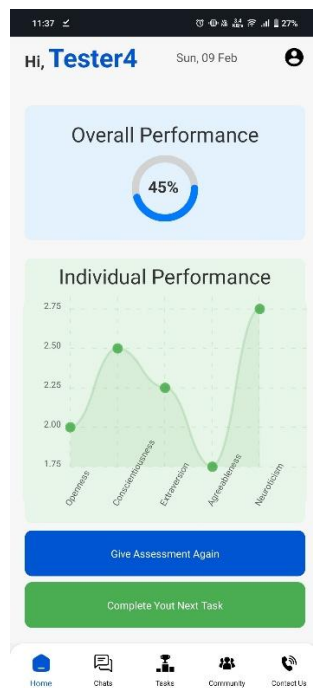
```



```

return res.status(400).send({
  success: false,
  message: 'Score is not calculated yet!',
});
}
return res.status(200).send({
  success: true,
  message: 'Score fetched sucessfully!',
  score: userScore,
});
} catch (error) {
return res.status(500).send({
  success: false,
  message: error.message,
});
}
};

```



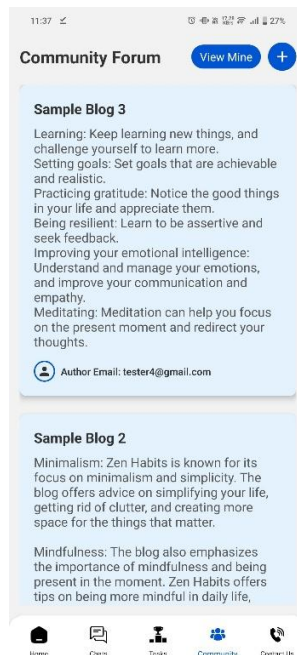
- **Community Engagement** to share experiences and participate in group activities.

➤ Algorithm for the Creating Post in Community ( JavaScript ):

```

const createpost = async (req, res) => {
  try {
    if (!req.body.title || !req.body.content) {
      return res.status(400).json({ error: 'Title and content are required' });
    }
    const post = await postModel({
      title: req.body.title,
      content: req.body.content,
      author: req.user.email,
    });
    post.save();
    return res.status(200).send({
      success: true,
      message: 'Post created sucessfully!',
    });
  } catch (error) {
    return res.status(500).send({
      success: false,
      message: error.message,
    });
  }
};

```



- **Task Generation and Completion** flow to Generate a task dynamically for a user regarding its level & constraints.

➤ Algorithm for the Task Generation & Completion ( JavaScript ):

```
const getNextTask = async (req, res) => {
  try {
    const orderList = ['openness', 'conscientiousness', 'extraversion', 'agreeableness',
      'neuroticism'];
    const userEmail = req.user.email;

    const existingPendingTask = await userTasksModel.findOne({ userEmail, status: 'pending'
      }).populate('taskId');

    if (existingPendingTask) {
      return res.status(200).send({
        success: true,
        message: 'You have an ongoing task!',
        task: existingPendingTask.taskId,
      });
    }

    const userScore = await scoreModel.findOne({ userEmail });
    if (!userScore) {
      return res.status(400).send({
        success: false,
        message: 'Score is not registered!',
      });
    }

    const scores = userScore.scores;
    const sumScore = Object.values(scores).reduce((acc, score) => acc + score, 0);

    let userLevel = '';
    if (sumScore >= 1 && sumScore <= 12) {
```

```
    userLevel = 'beginner';  
  } else if (sumScore >= 13 && sumScore <= 18) {  
    userLevel = 'intermediate';  
  } else if (sumScore >= 19 && sumScore <= 20) {  
    userLevel = 'advanced';  
  }  
}
```

```
const chosenTrait = orderList[Math.floor(Math.random() * orderList.length)];
```

```
const prevCompletedTasks = await userTasksModel.find({ userEmail, status: 'completed'  
  }).distinct('taskId');
```

```
const newTask = await TaskModel.findOne({  
  _id: { $nin: prevCompletedTasks },  
  level: userLevel,  
  trait: chosenTrait,  
});
```

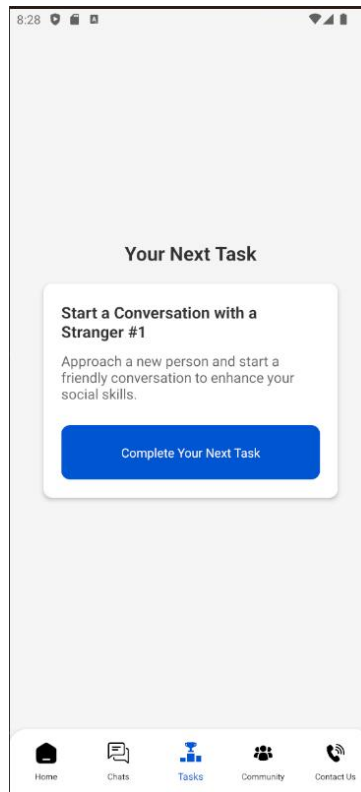
```
if (!newTask) {  
  return res.status(404).send({  
    success: false,  
    message: 'No suitable task found for the user!',  
  });  
}
```

```
const assignedTask = new userTasksModel({ userEmail, taskId: newTask._id });  
await assignedTask.save();
```

```
return res.status(200).send({  
  success: true,  
  message: 'Task successfully fetched!',  
  task: newTask,  
});  
} catch (error) {  
  return res.status(500).send({  
    success: false,  
    message: error.message,
```

```
});  
}  
};
```

```
const completeTask = async (req, res) => {  
  try {  
    const userEmail = req.user.email;  
  
    const task = await userTasksModel.findOne({ userEmail, status: 'pending' });  
  
    if (!task) {  
      return res.status(400).send({  
        success: false,  
        message: 'No pending task found for this user!',  
      });  
    }  
    task.status = 'completed';  
    await task.save();  
  
    return res.status(200).send({  
      success: true,  
      message: 'Task marked as completed!',  
    });  
  } catch (error) {  
    return res.status(500).send({  
      success: false,  
      message: error.message,  
    });  
  }  
};
```



Some Additional Features Samples :

A screenshot of a mobile application interface showing the "Edit Profile" screen. The status bar at the top shows the time 11:37 and various icons. The screen has a light gray background. At the top, the title "Edit Profile" is displayed in bold. Below the title, there are four input fields: "Name:" with the text "Tester4", "Email:" with the text "tester4@gmail.com", "Gender:" with the text "Male", and "Age:" with the text "21". At the bottom of the form is a blue button with the text "Save Changes".

A screenshot of a mobile application interface showing the "Feedback Form" screen. The status bar at the top shows the time 11:37 and various icons. The screen has a light gray background. At the top, the title "Feedback Form" is displayed in bold. Below the title, there are three sections: "Rate Us:" with five yellow stars, "Feedback Type:" with a dropdown menu showing "Select Type", and "Comments:" with a text area containing the placeholder "Write your comments here...". At the bottom of the form is a blue button with the text "Submit". Below the form, there is a navigation bar with five icons: Home, Chats, Tasks, Community, and Contact Us. The "Tasks" icon is highlighted in blue.

## **Procedure**

Participants began by completing an initial **self-assessment questionnaire**, which categorized their personality traits and identified areas for improvement. Based on this, they were assigned a structured learning path with **interactive exercises, and community participation**.

Users engaged with the app, completing tasks at their own pace. Progress was tracked through **in-app analytics**.

The study was conducted **online**, with participants using the app individually. Engagement data and behavioural changes were monitored to assess the effectiveness of the structured development approach.

---

## **Results**