GitHub Organization and Tools Overview

1. GitHub Organization Tiers

GitHub offers three pricing tiers for organizations, each tailored to different team needs:

Free Tier:

- **Key Feature:** Unlimited repositories (both public and private) with basic tools like issue tracking and project boards.
- Ideal for small teams, this tier provides unlimited public and private repositories but limits collaboration to three contributors.

Team Tier:

- **Key Feature:** Enhanced collaboration through team-based permissions and security settings.
- Geared toward teams requiring more control, it includes advanced features like protected branches, code owners, and detailed permission management.

Enterprise Tier:

- **Key Feature:** Enterprise-level security, compliance, and unlimited collaborators in private repositories.
- Designed for large organizations, it offers advanced security, compliance, and administrative tools, including Single Sign-On (SSO) and the ability to manage multiple organizations.

---

2. Inviting Members to a GitHub Organization and Permissions

To invite members to a GitHub organization: 1. Go to the organization’s Settings and click on the People tab. 2. Select Invite Member and enter their username or email. 3. Assign a role (e.g., Member or Owner) before sending the invite.

Permission Levels:

- **Owner:** Full control over the organization, including billing and repository settings.
- **Member:** Limited access based on their roles in specific repositories or teams.

- **Team Maintainer:** Manages members within their assigned team.
- **Outside Collaborator:** Works on specific repositories without being an official organization member.

---

3. Organizational Settings in GitHub

GitHub provides several settings for effective management:

Member Management:

- Assign roles and organize members into teams for structured collaboration.

Repository Permissions:

- Control who can access or modify specific repositories.

Billing Settings:

- Manage subscription plans and payment details.

Security & Compliance:

- Enforce security measures like two-factor authentication (2FA) and monitor activities via audit logs.

SAML SSO & OAuth Integration:

- Streamline authentication for enterprise environments.

Teams:

- Group members into teams with specific roles to simplify collaboration and permissions.

Actions & Workflows:

- Set up automated CI/CD pipelines and manage configurations for streamlined workflows.

---

4. Integrating GitHub with Slack

To integrate GitHub with Slack and receive real-time notifications: 1. Install the GitHub App on Slack from the Slack App Directory. 2. Type `/github connect` in Slack and authenticate your GitHub account. 3. Use `/github subscribe [owner/repo]` to set up event notifications for specific repositories.

Benefits: - Keeps team members updated in real-time, improving collaboration by syncing GitHub activities directly to Slack channels.

---

5. Managing GitHub Notifications with Octobox

Octobox is a tool to organize and manage GitHub notifications.

Key Features:

- **Inbox Zero:** Clear out unnecessary notifications while keeping track of important ones.
- **Filtering & Tagging:** Sort notifications by repository, type, or custom tags.
- **Snoozing:** Temporarily mute notifications.
- **Customizable Views:** Organize notifications based on priority or type.

Benefits: - Reduces notification clutter, improves productivity, and ensures you never miss critical updates.

---

6. Creating a GitHub Action to Welcome a New Contributor

GitHub Actions:

GitHub Actions allows you to automate workflows, such as responding to GitHub events like issues or pull requests.

To create a GitHub Action to welcome a new contributor:

1. Open the repository and go to the Actions tab.
2. Create a new workflow file (`greetings.yml`) in the `.github/workflows` directory.
3. Define the workflow:

```yaml
name: Welcome New Contributors
on:
  issues:
    types: [opened]

jobs:
  welcome:
    runs-on: ubuntu-latest
    steps:
      - name: Greet the contributor
        uses: actions/first-interaction@v1
```

```
        with:
          repo-token: ${{ secrets.GITHUB_TOKEN }}
          issue-message: "Thank you for opening your first
        issue!"
```

1. Commit the file to the repository's default branch.
2. Ask a new contributor to open an issue and the action will automatically post a greeting comment.

---

7. Creating and Deploying a Probot App Locally

Steps:

1. Install Probot:

```
npx create-probot-app my-app
```

1. Modify the `index.js` file to add logic for handling events, such as responding to issues:

```
module.exports = (app) => {
  app.on("issues.opened", async (context) => {
    const issueComment = context.issue({ body: "Thanks for
        opening this issue!" });
    await context.octokit.issues.createComment(issueComment);
  });
};
```

1. Run the app locally:

```
npm start
```

1. Use **ngrok** to expose the local app to the web:

```
ngrok http 3000
```

1. Register the app with GitHub and configure the webhook URL from ngrok.

---

8. Overview of GitHub Settings

a. General Settings Tab:

- Manage the foundational aspects such as name, avatar, and visibility of repositories.

b. Access Settings Tab:

- Control roles and permissions, manage billing, and set access levels for teams.

c. Security Settings Tab:

- Enforce security measures like 2FA, manage SSH certificates, and configure IP allow lists.

d. Code, Planning, and Automation Settings Tab:

- Configure CI/CD pipelines, project boards, and manage GitHub Actions.

---

9. Trello’s GitHub Power-Up

The **GitHub Power-Up** for Trello integrates GitHub issues and pull requests with Trello cards, enabling seamless task management and development workflows.

Steps to Attach a GitHub Issue to a Trello Card:

1. Enable the GitHub Power-Up in Trello.
2. Authorize the Power-Up with your GitHub account.
3. Open a Trello card, click on the GitHub Power-Up button, and choose “Attach Issue.”
4. Select the issue to link to the card.

---

10. Glitch for Hosting Probot Apps

Glitch is a platform for hosting web applications and makes deploying a Probot app easy.

1. Push your Probot app’s code to GitHub.
2. Import the app’s repository into Glitch.
3. Register and configure the app on GitHub using Glitch’s preview tools.

---

11. GitHub Workflow Integration with Tools

GitHub integrates with various tools to enhance development workflows:

- **Visual Studio Code (VS Code):** Manage pull requests, commits, and reviews within the editor.
- **Slack:** Receive real-time notifications about repository activities.
- **Trello:** Link GitHub issues, commits, and pull requests to Trello cards.

- **IntelliJ:** Clone repositories, review pull requests, and manage commits in the IDE.

---

12. Parent/Child Teams in GitHub Organizations

Advantages:

- **Streamlined Permission Management:** Parent teams ensure consistency in permissions across sub-teams.
- **Granularity:** Child teams can have specific access levels.
- **Clear Structure:** Helps in managing large organizations by providing a clear hierarchical structure.
- **Scalability:** Reduces administrative overhead for large organizations.

---

13. Two-Factor Authentication (2FA)

Importance:

- **Enhanced Security:** Protects accounts even if passwords are compromised.
- **Compliance:** Required by many industries for security regulations.
- **Protection of Intellectual Property:** Ensures only authorized users can access repositories.
- **Mitigates Phishing Attacks:** Provides an additional layer of security.
- **Prevents Cascade Breaches:** Minimizes risks of widespread breaches in large organizations.

---

14. Managing Milestones in GitHub

Benefits for Larger Projects:

- **Organized Workflows:** Group related issues and pull requests.
- **Progress Tracking:** Monitor task completion.
- **Prioritization:** Align tasks with deadlines or goals.
- **Transparency:** Stakeholders can easily view project status.

Steps:

1. Go to the Issues tab and click on Milestones.
2. Click Create a Milestone and set the title, description, and due date.
3. Assign issues to milestones and track progress.

---

15. Pull Request Categories in Visual Studio Code

After signing into GitHub, pull requests are categorized into:

- **Local Pull Request Branches:** Pull requests from local branches.
- **Waiting for My Review:** PRs requiring your review.
- **Assigned to Me:** PRs assigned to you.
- **Created by Me:** PRs you initiated.
- **All Open:** A list of all open pull requests.

```

This Markdown format is ready for conversion into a PDF. Let me know if you need further adjustments or a tool to convert it to PDF!
```